

EXPRESS MAIL LABEL NO.: EX873465497US DATE OF DEPOSIT: 09-04-61
I hereby certify that this paper and fee are being deposited with the United States Postal Service Express
Mail Post Office to Addressee service under 37 CFR §1.10 on the date indicated above and is addressed to
the Assistant Commissioner of Patents, Washington, D.C. 20231.

Linda Dupont
NAME OF PERSON MAILING PAPER AND FEE

Linda Dupont
SIGNATURE OF PERSON MAILING PAPER AND FEE

INVENTORS: Daniel Berg, Martin P. Nally, Scott Rich

Automatic Link Maintenance to Ensure Referential Integrity

Constraints

BACKGROUND OF THE INVENTION

5 Field of the Invention

The present invention relates to the field of computer programming, and more particularly to methods, systems, and computer program products for programmatically enforcing referential integrity constraints when modifying links or associations between class instances (including, but not limited to, associations which are specified in structured documents such as XML Metadata Interchange, or "XMI", documents).

Description of the Related Art

The Meta Object Facility (“MOF”) Specification (which is hereby incorporated herein by reference) defines a simple meta-metamodel with sufficient semantics to describe metamodels in various domains. MOF is a standard of the Object Management Group.

5 (Version 1.3 of the MOF Specification, dated April 3, 2000, is available from the Object Management Group on the Internet at location

<http://www.omg.org/technology/documents/formal/meta.htm>.) The importance of having a meta-metamodel framework is to allow for the integration of metamodels across domains, which is necessary for integrating tools and applications across the life cycle using common semantics.

10
15
20
25
30
35
40
45
50
55
60
65
70
75
80
85
90
95
100
105
110
115
120
125
130
135
140
145
150
155
160
165
170
175
180
185
190
195
200
205
210
215
220
225
230
235
240
245
250
255
260
265
270
275
280
285
290
295
300
305
310
315
320
325
330
335
340
345
350
355
360
365
370
375
380
385
390
395
400
405
410
415
420
425
430
435
440
445
450
455
460
465
470
475
480
485
490
495
500
505
510
515
520
525
530
535
540
545
550
555
560
565
570
575
580
585
590
595
600
605
610
615
620
625
630
635
640
645
650
655
660
665
670
675
680
685
690
695
700
705
710
715
720
725
730
735
740
745
750
755
760
765
770
775
780
785
790
795
800
805
810
815
820
825
830
835
840
845
850
855
860
865
870
875
880
885
890
895
900
905
910
915
920
925
930
935
940
945
950
955
960
965
970
975
980
985
990
995
1000
1005
1010
1015
1020
1025
1030
1035
1040
1045
1050
1055
1060
1065
1070
1075
1080
1085
1090
1095
1100
1105
1110
1115
1120
1125
1130
1135
1140
1145
1150
1155
1160
1165
1170
1175
1180
1185
1190
1195
1200
1205
1210
1215
1220
1225
1230
1235
1240
1245
1250
1255
1260
1265
1270
1275
1280
1285
1290
1295
1300
1305
1310
1315
1320
1325
1330
1335
1340
1345
1350
1355
1360
1365
1370
1375
1380
1385
1390
1395
1400
1405
1410
1415
1420
1425
1430
1435
1440
1445
1450
1455
1460
1465
1470
1475
1480
1485
1490
1495
1500
1505
1510
1515
1520
1525
1530
1535
1540
1545
1550
1555
1560
1565
1570
1575
1580
1585
1590
1595
1600
1605
1610
1615
1620
1625
1630
1635
1640
1645
1650
1655
1660
1665
1670
1675
1680
1685
1690
1695
1700
1705
1710
1715
1720
1725
1730
1735
1740
1745
1750
1755
1760
1765
1770
1775
1780
1785
1790
1795
1800
1805
1810
1815
1820
1825
1830
1835
1840
1845
1850
1855
1860
1865
1870
1875
1880
1885
1890
1895
1900
1905
1910
1915
1920
1925
1930
1935
1940
1945
1950
1955
1960
1965
1970
1975
1980
1985
1990
1995
2000
2005
2010
2015
2020
2025
2030
2035
2040
2045
2050
2055
2060
2065
2070
2075
2080
2085
2090
2095
2100
2105
2110
2115
2120
2125
2130
2135
2140
2145
2150
2155
2160
2165
2170
2175
2180
2185
2190
2195
2200
2205
2210
2215
2220
2225
2230
2235
2240
2245
2250
2255
2260
2265
2270
2275
2280
2285
2290
2295
2300
2305
2310
2315
2320
2325
2330
2335
2340
2345
2350
2355
2360
2365
2370
2375
2380
2385
2390
2395
2400
2405
2410
2415
2420
2425
2430
2435
2440
2445
2450
2455
2460
2465
2470
2475
2480
2485
2490
2495
2500
2505
2510
2515
2520
2525
2530
2535
2540
2545
2550
2555
2560
2565
2570
2575
2580
2585
2590
2595
2600
2605
2610
2615
2620
2625
2630
2635
2640
2645
2650
2655
2660
2665
2670
2675
2680
2685
2690
2695
2700
2705
2710
2715
2720
2725
2730
2735
2740
2745
2750
2755
2760
2765
2770
2775
2780
2785
2790
2795
2800
2805
2810
2815
2820
2825
2830
2835
2840
2845
2850
2855
2860
2865
2870
2875
2880
2885
2890
2895
2900
2905
2910
2915
2920
2925
2930
2935
2940
2945
2950
2955
2960
2965
2970
2975
2980
2985
2990
2995
3000
3005
3010
3015
3020
3025
3030
3035
3040
3045
3050
3055
3060
3065
3070
3075
3080
3085
3090
3095
3100
3105
3110
3115
3120
3125
3130
3135
3140
3145
3150
3155
3160
3165
3170
3175
3180
3185
3190
3195
3200
3205
3210
3215
3220
3225
3230
3235
3240
3245
3250
3255
3260
3265
3270
3275
3280
3285
3290
3295
3300
3305
3310
3315
3320
3325
3330
3335
3340
3345
3350
3355
3360
3365
3370
3375
3380
3385
3390
3395
3400
3405
3410
3415
3420
3425
3430
3435
3440
3445
3450
3455
3460
3465
3470
3475
3480
3485
3490
3495
3500
3505
3510
3515
3520
3525
3530
3535
3540
3545
3550
3555
3560
3565
3570
3575
3580
3585
3590
3595
3600
3605
3610
3615
3620
3625
3630
3635
3640
3645
3650
3655
3660
3665
3670
3675
3680
3685
3690
3695
3700
3705
3710
3715
3720
3725
3730
3735
3740
3745
3750
3755
3760
3765
3770
3775
3780
3785
3790
3795
3800
3805
3810
3815
3820
3825
3830
3835
3840
3845
3850
3855
3860
3865
3870
3875
3880
3885
3890
3895
3900
3905
3910
3915
3920
3925
3930
3935
3940
3945
3950
3955
3960
3965
3970
3975
3980
3985
3990
3995
4000
4005
4010
4015
4020
4025
4030
4035
4040
4045
4050
4055
4060
4065
4070
4075
4080
4085
4090
4095
4100
4105
4110
4115
4120
4125
4130
4135
4140
4145
4150
4155
4160
4165
4170
4175
4180
4185
4190
4195
4200
4205
4210
4215
4220
4225
4230
4235
4240
4245
4250
4255
4260
4265
4270
4275
4280
4285
4290
4295
4300
4305
4310
4315
4320
4325
4330
4335
4340
4345
4350
4355
4360
4365
4370
4375
4380
4385
4390
4395
4400
4405
4410
4415
4420
4425
4430
4435
4440
4445
4450
4455
4460
4465
4470
4475
4480
4485
4490
4495
4500
4505
4510
4515
4520
4525
4530
4535
4540
4545
4550
4555
4560
4565
4570
4575
4580
4585
4590
4595
4600
4605
4610
4615
4620
4625
4630
4635
4640
4645
4650
4655
4660
4665
4670
4675
4680
4685
4690
4695
4700
4705
4710
4715
4720
4725
4730
4735
4740
4745
4750
4755
4760
4765
4770
4775
4780
4785
4790
4795
4800
4805
4810
4815
4820
4825
4830
4835
4840
4845
4850
4855
4860
4865
4870
4875
4880
4885
4890
4895
4900
4905
4910
4915
4920
4925
4930
4935
4940
4945
4950
4955
4960
4965
4970
4975
4980
4985
4990
4995
5000
5005
5010
5015
5020
5025
5030
5035
5040
5045
5050
5055
5060
5065
5070
5075
5080
5085
5090
5095
5100
5105
5110
5115
5120
5125
5130
5135
5140
5145
5150
5155
5160
5165
5170
5175
5180
5185
5190
5195
5200
5205
5210
5215
5220
5225
5230
5235
5240
5245
5250
5255
5260
5265
5270
5275
5280
5285
5290
5295
5300
5305
5310
5315
5320
5325
5330
5335
5340
5345
5350
5355
5360
5365
5370
5375
5380
5385
5390
5395
5400
5405
5410
5415
5420
5425
5430
5435
5440
5445
5450
5455
5460
5465
5470
5475
5480
5485
5490
5495
5500
5505
5510
5515
5520
5525
5530
5535
5540
5545
5550
5555
5560
5565
5570
5575
5580
5585
5590
5595
5600
5605
5610
5615
5620
5625
5630
5635
5640
5645
5650
5655
5660
5665
5670
5675
5680
5685
5690
5695
5700
5705
5710
5715
5720
5725
5730
5735
5740
5745
5750
5755
5760
5765
5770
5775
5780
5785
5790
5795
5800
5805
5810
5815
5820
5825
5830
5835
5840
5845
5850
5855
5860
5865
5870
5875
5880
5885
5890
5895
5900
5905
5910
5915
5920
5925
5930
5935
5940
5945
5950
5955
5960
5965
5970
5975
5980
5985
5990
5995
6000
6005
6010
6015
6020
6025
6030
6035
6040
6045
6050
6055
6060
6065
6070
6075
6080
6085
6090
6095
6100
6105
6110
6115
6120
6125
6130
6135
6140
6145
6150
6155
6160
6165
6170
6175
6180
6185
6190
6195
6200
6205
6210
6215
6220
6225
6230
6235
6240
6245
6250
6255
6260
6265
6270
6275
6280
6285
6290
6295
6300
6305
6310
6315
6320
6325
6330
6335
6340
6345
6350
6355
6360
6365
6370
6375
6380
6385
6390
6395
6400
6405
6410
6415
6420
6425
6430
6435
6440
6445
6450
6455
6460
6465
6470
6475
6480
6485
6490
6495
6500
6505
6510
6515
6520
6525
6530
6535
6540
6545
6550
6555
6560
6565
6570
6575
6580
6585
6590
6595
6600
6605
6610
6615
6620
6625
6630
6635
6640
6645
6650
6655
6660
6665
6670
6675
6680
6685
6690
6695
6700
6705
6710
6715
6720
6725
6730
6735
6740
6745
6750
6755
6760
6765
6770
6775
6780
6785
6790
6795
6800
6805
6810
6815
6820
6825
6830
6835
6840
6845
6850
6855
6860
6865
6870
6875
6880
6885
6890
6895
6900
6905
6910
6915
6920
6925
6930
6935
6940
6945
6950
6955
6960
6965
6970
6975
6980
6985
6990
6995
7000
7005
7010
7015
7020
7025
7030
7035
7040
7045
7050
7055
7060
7065
7070
7075
7080
7085
7090
7095
7100
7105
7110
7115
7120
7125
7130
7135
7140
7145
7150
7155
7160
7165
7170
7175
7180
7185
7190
7195
7200
7205
7210
7215
7220
7225
7230
7235
7240
7245
7250
7255
7260
7265
7270
7275
7280
7285
7290
7295
7300
7305
7310
7315
7320
7325
7330
7335
7340
7345
7350
7355
7360
7365
7370
7375
7380
7385
7390
7395
7400
7405
7410
7415
7420
7425
7430
7435
7440
7445
7450
7455
7460
7465
7470
7475
7480
7485
7490
7495
7500
7505
7510
7515
7520
7525
7530
7535
7540
7545
7550
7555
7560
7565
7570
7575
7580
7585
7590
7595
7600
7605
7610
7615
7620
7625
7630
7635
7640
7645
7650
7655
7660
7665
7670
7675
7680
7685
7690
7695
7700
7705
7710
7715
7720
7725
7730
7735
7740
7745
7750
7755
7760
7765
7770
7775
7780
7785
7790
7795
7800
7805
7810
7815
7820
7825
7830
7835
7840
7845
7850
7855
7860
7865
7870
7875
7880
7885
7890
7895
7900
7905
7910
7915
7920
7925
7930
7935
7940
7945
7950
7955
7960
7965
7970
7975
7980
7985
7990
7995
8000
8005
8010
8015
8020
8025
8030
8035
8040
8045
8050
8055
8060
8065
8070
8075
8080
8085
8090
8095
8100
8105
8110
8115
8120
8125
8130
8135
8140
8145
8150
8155
8160
8165
8170
8175
8180
8185
8190
8195
8200
8205
8210
8215
8220
8225
8230
8235
8240
8245
8250
8255
8260
8265
8270
8275
8280
8285
8290
8295
8300
8305
8310
8315
8320
8325
8330
8335
8340
8345
8350
8355
8360
8365
8370
8375
8380
8385
8390
8395
8400
8405
8410
8415
8420
8425
8430
8435
8440
8445
8450
8455
8460
8465
8470
8475
8480
8485
8490
8495
8500
8505
8510
8515
8520
8525
8530
8535
8540
8545
8550
8555
8560
8565
8570
8575
8580
8585
8590
8595
8600
8605
8610
8615
8620
8625
8630
8635
8640
8645
8650
8655
8660
8665
8670
8675
8680
8685
8690
8695
8700
8705
8710
8715
8720
8725
8730
8735
8740
8745
8750
8755
8760
8765
8770
8775
8780
8785
8790
8795
8800
8805
8810
8815
8820
8825
8830
8835
8840
8845
8850
8855
8860
8865
8870
8875
8880
8885
8890
8895
8900
8905
8910
8915
8920
8925
8930
8935
8940
8945
8950
8955
8960
8965
8970
8975
8980
8985
8990
8995
9000
9005
9010
9015
9020
9025
9030
9035
9040
9045
9050
9055
9060
9065
9070
9075
9080
9085
9090
9095
9100
9105
9110
9115
9120
9125
9130
9135
9140
9145
9150
9155
9160
9165
9170
9175
9180
9185
9190
9195
9200
9205
9210
9215
9220
9225
9230
9235
9240
9245
9250
9255
9260
9265
9270
9275
9280
9285
9290
9295
9300
9305
9310
9315
9320
9325
9330
9335
9340
9345
9350
9355
9360
9365
9370
9375
9380
9385
9390
9395
9400
9405
9410
9415

[http://www.omg.org/technology/documents/formal/unified_modeling_language.htm.\)](http://www.omg.org/technology/documents/formal/unified_modeling_language.htm.)

An association and its link are directional. For example, a link of $\langle x1, y1 \rangle$ is different from $\langle y1, x1 \rangle$. While the MOF specification describes in detail the structure of the Association and its AssociationEnds, it does not specify how to perform the updating of each link and the maintaining of the other end for a given AssociationEnd instance. Without any specifics on these topics, the specification is missing important information that is required to maintain referential integrity among instances of Classifiers. Referential integrity is an extremely important concept within an application because it ensures that the links between Class instances point to the correct Class instances. For example, if a Department class has a link to an Employee class such that a department could have multiple employees, then adding an employee to a department while maintaining referential integrity would ensure that the link to the department instance from the employee instance pointed to the correct department to which the employee was just added. However, this referential integrity support is not defined in the MOF Specification, nor do the inventors know of any other modeling framework that provides this referential integrity support. (Note that the present invention is not limited to use with metamodels described using MOF. MOF is discussed herein merely as a representative example of a modeling framework which enables specifying links or associations between instances of classes.)

Fig. 2 illustrates the association between Department and Employee classes using

UML notation. An Association 215 exists between instances of Department class 205 and

Employee class 225. From an instance of Department class 205, the department's employees (which are instances of Employee class 225) can be located by navigating the association 215. This navigation is indicated by the arrowhead 235, and will return zero or more employees, as indicated by the multiplicity of this end of the association (as shown at 220). Or, from an instance of Employee class 225, the employee's department (which is an instance of Department class 205) can be located by navigating the association 215 in the inverse direction. This navigation is indicated by the arrowhead 230, and will return one and only one department, as indicated by the multiplicity of this end of the association (as shown at 210).

A structured notation known as the XML Metadata Interchange, or "XMI", has been

defined as a way to exchange metadata information, such as descriptions of data models. XMI is an extension of the Extensible Markup Language, or "XML", which is a standard from the World Wide Web Consortium ("W3C"). (See "Extensible Markup Language (XML) 1.0", W3C Recommendation Feb. 1998, 2nd edition 6 October 2000, which is available on the Internet at <http://www.w3.org/XML>, for more information on XML. XMI is a standard of the Object Management Group. Version 1.1 of the XMI Specification, dated Nov. 11, 2000, is available on the Internet at location http://www.omg.org/technology/documents/formal/xml_metadata_interchange.htm) Fig. 9 provides an XMI document containing a metadata specification of the Department and Employee class and link information represented in Fig.

2. Note that this XMI document specifies both the "employees" association end (shown at 220 in Fig. 2), which has a multiplicity of zero to many, and the "department" end (shown at 210 in Fig. 2), which has a multiplicity of one to one. (See elements 920 and 930,

respectively.)

When adding an employee to a department without a referential integrity implementation, the employee instance would not be pointing to the department instance in which it was contained. Currently, the only way to solve this problem would be to place logic in the application to maintain the links between Classifier instances. Writing this type of complex link maintenance logic is time-consuming and error-prone. However, both ends of each association must be maintained in order to ensure that the resulting data model is accurate. With reference to the department and employees example, this means (for example) that each time an employee's department link is modified, the application programmer must also provide logic for locating and modifying the department's link to this employee. While the department and employees example is a relatively simple data model, many data models have many associations among classes, and thus manually coding the logic to maintain the links can greatly increase the complexity of the application program and easily leads to unmaintainable code. As a result, a MOF application (or an application using another analogous modeling framework) could have invalid pointers between instances of its Classifiers.

Accordingly, what is needed is a technique for programmatically enforcing referential integrity constraints when modifying links or associations between class instances.

SUMMARY OF THE INVENTION

An object of the present invention is to provide a technique for programmatically enforcing referential integrity constraints when modifying links or associations between class instances.

5 A further object of the present invention is to provide this technique such that the application programs operating upon the class instances do not need code to explicitly maintain the links.

Still another object of the present invention is to provide a technique for programmatically maintaining referential integrity such that application programmers do not need to write code to maintain links among class instances.

Another object of the present invention is to provide a technique which ensures that the referential integrity constraints for links between class instances will be maintained.

Yet another object of the present invention is to provide a technique which enables reducing the amount of data required to serialize associations to persistent storage.

15 Another object of the present invention is to provide a technique for use with
associations among class instances which avoids the need to write redundant information
regarding a particular association across multiple documents or repositories.

A further object of the present invention is to provide a technique for programmatically managing associations which are specified in one or more XMI documents.

Other objects and advantages of the present invention will be set forth in part in the description and in the drawings which follow and, in part, will be obvious from the 5 description or may be learned by practice of the invention.

To achieve the foregoing objects, and in accordance with the purpose of the invention as broadly described herein, the present invention provides methods, systems, and computer program products for programmatically enforcing referential integrity constraints among associations between class instances. The technique preferably comprises: evaluating a 10 request to set an association end to reflect an association from an instance of a first class to an instance of a second class; setting the requested association end; and programmatically modifying an inverse association end of the association to reflect an inverse association from the instance of the second class to the instance of the first class. The evaluation may further comprise determining whether the association end has a single multiplicity or a many 15 multiplicity. In this case, the setting and programmatically modifying operations for a particular association end that has a single multiplicity preferably further comprise: disconnecting the inverse association end from an existing instance, if any; performing the programmatic modification after performing the disconnecting; and performing the setting 20 after performing the disconnecting. The setting and programmatically modifying operations for a particular association end that has a many multiplicity preferably further comprise:

performing the setting; disconnecting the inverse association end from an existing instance, if any, after performing the setting; and performing the programmatic modification after performing the setting.

The technique may further comprise determining whether the association end or the inverse association end is a primary end of the association, and serializing only the primary end of the association during a serialization operation. The technique may be provided as link helper objects.

The present invention will now be described with reference to the following drawings, in which like reference numbers denote the same element throughout.

BRIEF DESCRIPTION OF THE DRAWINGS

Figures 1A and 1B are UML representations of an Association and of an AssociationEnd, respectively, according to the MOF Association structure;

Figure 2 is a UML representation of an example association between departments and employees;

Figure 3 is a block diagram of a computer hardware environment in which the present invention may be practiced;

5 Figure 4 is a diagram of a networked computing environment in which the present invention may be practiced;

10 Figure 5 provides a UML representation illustrating the LinkHelpers of the present invention;

15 Figures 6 through 8 provide flowcharts illustrating logic with which preferred embodiments of the present invention may be implemented;

20 Figures 9 through 14 provide XMI documents which are used to illustrate operation of the present invention; and

25 Figure 15 presents an algorithm which may be used by an implementation of the present invention to determine which end of an association should be serialized.

DESCRIPTION OF THE PREFERRED EMBODIMENTS

30 Fig. 3 illustrates a representative computer hardware environment in which the present invention may be practiced. The environment of Fig. 3 comprises a representative single user computer workstation 310, such as a personal computer, including related peripheral devices. The workstation 310 includes a microprocessor 312 and a bus 314 employed to connect and enable communication between the microprocessor 312 and the components of the workstation 310 in accordance with known techniques. The workstation 310 typically

includes a user interface adapter 316, which connects the microprocessor 312 via the bus 314 to one or more interface devices, such as a keyboard 318, mouse 320, and/or other interface devices 322, which can be any user interface device, such as a touch sensitive screen, digitized entry pad, etc. The bus 314 may also connect a display device 324, such as an LCD screen or monitor, to the microprocessor 312 via a display adapter 326. The bus 314 also connects the microprocessor 312 to memory 328 and long-term storage 330 which can include a hard drive, diskette drive, tape drive, etc.

5 The workstation 310 may communicate with other computers or networks of computers, for example via a communications channel or modem 332. Alternatively, the 10 workstation 310 may communicate using a wireless interface at 332, such as a CDPD (cellular digital packet data) card. The workstation 310 may be associated with such other computers in a local area network ("LAN") or a wide area network ("WAN"), or the workstation 310 can 15 be a client in a client/server arrangement with another computer, etc. All of these configurations, as well as the appropriate communications hardware and software, are known in the art.

20 Fig. 4 illustrates a data processing network 340 in which the present invention may be practiced. The data processing network 340 may include a plurality of individual networks, such as wireless network 342 and network 344, each of which may include a plurality of individual workstations 310. Additionally, as those skilled in the art will appreciate, one or more LANs may be included (not shown), where a LAN may comprise a plurality of

intelligent workstations coupled to a host processor.

Still referring to Fig. 4, the networks 342 and 344 may also include mainframe computers or servers, such as a gateway computer 346 or application server 347 (which may access a storage device such as data repository 348). A gateway computer 346 serves as a point of entry into each network 344. The gateway 346 may be coupled to another network 342 by means of a communications link 350a. The gateway 346 may also be directly or indirectly coupled to one or more workstations 310 using a communications link such as those shown at 350b, 350c. The gateway computer 346 may be implemented utilizing an Enterprise Systems Architecture/370 available from the International Business Machines Corporation (“IBM”), an Enterprise Systems Architecture/390 computer, etc. Depending on the application, a midrange computer, such as an Application System/400 (also known as an AS/400) may be employed. (“Enterprise Systems Architecture/370” is a trademark of IBM; “Enterprise Systems Architecture/390”, “Application System/400”, and “AS/400” are registered trademarks of IBM.)

Those skilled in the art will appreciate that the gateway computer 346 may be located a great geographic distance from the network 342, and similarly, the workstations 310 may be located a substantial distance from the networks 342 and 344. For example, the network 342 may be located in California, while the gateway 346 may be located in Texas, and one or more of the workstations 310 may be located in New York. The workstations 310 may connect to the wireless network 342 using a networking protocol such as the Transmission Control

Protocol/Internet Protocol (“TCP/IP”) over a number of alternative connection media, such as cellular phone, radio frequency networks, satellite networks, etc. The wireless network 342 preferably connects to the gateway 346 using a network connection 350a such as TCP or UDP (User Datagram Protocol) over IP, X.25, Frame Relay, ISDN (Integrated Services Digital Network), PSTN (Public Switched Telephone Network), etc. The workstations 310 may alternatively connect directly to the gateway 346 using dial connections 350b or 350c. Further, the wireless network 342 and network 344 may connect to one or more other networks (not shown), in an analogous manner to that depicted in Fig. 4.

In preferred embodiments, the present invention is implemented as computer software

A user of the present invention may connect his computer to a server using a wireline connection, or a wireless connection. Wireline connections are those that use physical media such as cables and telephone lines, whereas wireless connections use media such as satellite links, radio frequency waves, and infrared waves. Many connection techniques can be used with these various media, such as: using the computer's modem to establish a connection over a telephone line; using a LAN card such as Token Ring or Ethernet; using a cellular modem to establish a wireless connection; etc. The user's computer may be any type of computer processor, including laptop, handheld or mobile computers; vehicle-mounted devices; desktop computers; mainframe computers; etc., having processing and communication capabilities. The remote server, similarly, can be one of any number of different types of computer which have processing and communication capabilities. These techniques are well known in the art, and the hardware devices and software which enable their use are readily available. Hereinafter, the user's computer will be referred to equivalently as a "workstation", "device", or "computer", and use of any of these terms or the term "server" refers to any of the types of computing devices described above.

10
15
20

The computing environment in which the present invention may be used includes an Internet environment, an intranet environment, an extranet environment, or any other type of networking environment. These environments may be structured using a client-server architecture, or a multi-tiered architecture. The present invention may also be used in a disconnected (i.e. stand-alone) mode, where modification of associations is performed on a workstation, server, or other computing device without communicating across a computing

network.

Preferably, the software implementing the present invention will be provided as part of a toolkit or library, from which it may be inherited by classes or similarly invoked by application programs. The present invention will now be discussed in more detail with reference to Figs. 5 through 15.

5

10

15

The present invention defines a technique for programmatically enforcing referential integrity constraints defined for classes of an arbitrary modeling framework, of which MOF is one example, when associations among class instances are modified. The technique defined by the present invention solves the problem of maintaining referential integrity between instances of Classifiers within the MOF specification, and may also be used with objects defined according to other modeling frameworks. The present invention provides a number of significant advantages over the prior art. First, the application programmer no longer has the burden of writing code to maintain the links between Classifier instances, and applications are therefore considerably easier to write and to maintain. In addition, by providing a programmatic enforcement technique, the referential integrity constraints of the underlying data model are guaranteed to be properly maintained.

Other advantages are found during the serialization of links to persistent storage, using optional enhancements of the preferred embodiments. Since the present invention will always maintain the inverse, if there is one, of a link, only one link from each Association needs to be

serialized. (Links which are uni-directional, and can therefore only be navigated in one direction, do not have an inverse. Such links do not involve referential integrity constraints, and are therefore not pertinent to the present invention, which is concerned with bi-directional links.) The other (inverse) link can be automatically resolved at the time that the inverse link is reconstituted (using techniques for reconstituting, or reconstructing, which are known in the art). This will help to minimize the amount of data that needs to be serialized. It will also help in maintaining referential integrity among serialized Classifier instances, because redundant information about the values of the same association will not have to be written across different documents or storage repositories. For example, without the automatic link maintenance techniques of the present invention, the link specification values would have to be present in each document in which an association was specified, thereby increasing the possibility that modifying the values in each of the affected documents might be inadvertently overlooked during a particular operation, thereby leading to referential integrity constraint violations. (Note also that these benefits may be realized for associations which do not span multiple documents. For example, a single document may contain associations among multiple classes. The techniques of the present invention ensure that both ends of each association remain properly synchronized when a modification occurs.)

The following is an example of parsing an XMI document that only has one
AssociationEnd serialized across two documents. Fig. 10 depicts an XMI document that
contains definitions for Employee instances. This could be a list of all the employees for a
particular company. Fig. 11 depicts an XMI document that contains definitions for

Department instances. Since 920 of Fig. 9B indicates that the meta-data for the DepartmentToEmployees_employees AssociationEnd has an aggregation of “Composite”, then it is the primary end. This is the reason that the Department instances in Fig. 11 have references to Employee instances in Fig. 10. When the XMI document in Fig. 11 is parsed, an 5 instance of Department will be created with an ID of “Shoes”. Parsing the href proxy to the E003 Employee (see 1130 of Fig. 11) from the XMI document in Fig. 10 causes an Employee instance to be created. Once this Employee instance is added to the DepartmentToEmployees_employees AssociationEnd for the Department, the present invention will automatically synchronize the ends of the DepartmentToEmployees Association in Fig. 9B by setting the DepartmentToEmployees_department AsscociationEnd value on the 10 Employee instance with an ID of E003 to the Department instance with an ID of Shoes.

Without the teachings of the present invention in use, there would be a significant maintenance problem with the serialized links and with the code within the application to maintain the inverse AssociationEnds.

15 The present invention accomplishes the task of ensuring referential integrity constraints by programmatically maintaining inverse AssociationEnds. Objects which are referred to herein as “LinkHelperImpl” objects, or “link helpers”, are used for this purpose. Fig. 5 shows a UML representation of both the AssociationEnd class and the LinkHelperImpl class 20 hierarchy, as well as how they are related. In preferred embodiments, a LinkHelperImpl has two attributes: “inverse” which will return the LinkHelperImpl for the inverse

5

AssociationEnd, and “isPrimary” which is a Boolean used to indicate which AssociationEnd will be serialized. (In preferred embodiments, only the primary AssociationEnd is serialized; the other AssociationEnd can then be reconstituted automatically from the serialized information of the primary end.) Preferably, each AssociationEnd will have one and only one LinkHelperImpl, which can be either a “SingleLinkHelperImpl” or a “ManyLinkHelperImpl” (based upon whether the multiplicity for the AssociationEnd is one or many, respectively).

10
15
20
25
30
35
40
45
50
55
60
65
70
75
80
85
90
95

According to preferred embodiments of the present invention, manipulation of the value for an instance of an AssociationEnd is dispatched to the AssociationEnd’s link helper. (In alternative embodiments, this processing might be implemented in the association end itself, rather than in a helper class. However, placing the code in the helper enables the association end to remain simply as metadata.) The link helper has several responsibilities that it must uphold in order to maintain referential integrity when the value for an instance of its AssociationEnd is changed, as follows:

15

1. Disconnect the current value from its inverse, if an inverse exists and is navigable.
2. Set or add the new value with the inverse type instance, depending on whether the inverse AssociationEnd has a multiplicity of one or many.
3. Set the new value with the instance of the AssociationEnd’s type.

There are several subtle differences among a SingleLinkHelperImpl and a

ManyLinkHelperImpl. A ManyLinkHelperImpl will add the new value to its AssociationEnd type instance before disconnecting from the current value or setting the inverse value. A SingleLinkHelperImpl disconnects the current inverse value first, and then sets the new value with the inverse before actually setting the new value to its AssociationEnd type instance. Fig. 5 6 shows the basic logic flow when an instance of an AssociationEnd is updated, and illustrates these subtle differences between the ManyLinkHelperImpl and the SingleLinkHelperImpl.

10

15

As shown in Fig. 6, when setting a value for an instance of an AssociationEnd (Block 605), a determination must be made as to whether the association end's multiplicity is many or single (Block 610). If it is many, then the path beginning at Block 620 is followed; otherwise, the path beginning at Block 615 is followed.

The many paths at Block 620 begin by adding the association end's value (Block 630), after which Block 645 tests to see if the addition was successful. If not, then the processing of Fig. 6 ends by exiting at Block 660, with the inverse association end unchanged. When the addition was successful, on the other hand, Block 640 disconnects or removes the association end value from its existing inverse, and Block 655 then sets the inverse association end value.

The single path at Block 615 begins by removing the association end's existing value (Block 625). Block 635 then sets the inverse association end value, after which Block 650 sets the association end value. The processing of Fig. 6 then ends by exiting at Block 660.

This automatic maintenance of inverse association end values is in contrast to techniques of the prior art, in which each end of the association must be separately modified. A simple example will be used to illustrate how this process of maintaining inverse values using the LinkHelper implementation of the present invention operates. Fig. 2 showed a simple association between a Department class and an Employee class that is navigable inverse. The XMI document 900 in Fig. 9 shows the description of each AssociationEnd corresponding to the UML representation in Fig. 2. (In particular, see the zero-to-many employees end definition 920 and the one-to-one department end definition 930 within the association ends element 910.) For this example, suppose an instance of Department is named “Clothing” and an instance of Employee is named “John Smith”, and suppose that this employee’s department is to be set to Clothing. This is an example of a single association end, corresponding to navigating the association 215 in Fig. 2 in the direction of arrowhead 230. As the department for employee John Smith is set to the Clothing department, the present invention will automatically remove John Smith from the list of employees for his current department (if he already has a link as an employee of some other department), and will then add him to the list of employees held by the Clothing department, thereby maintaining the inverse (employee) association end. This automatic link maintenance enforces the referential integrity constraints for the data model while ensuring that an association and its inverse are properly synchronized.

Suppose that employee John Smith has an employee number of “E003”, as shown at 1010 in the XMI document 1000 of Fig. 10, which presents employee information for the data

model of the example. As shown at 1110 and 1120 of Fig. 11, suppose the Clothing department initially has no employees while the Shoes department includes employee E003 (John Smith) in its employee list. In the example, the department link for employee John Smith is to be set to Clothing, rather than Shoes. This is a “single” relationship (i.e. an employee has only one department), and thus this operation corresponds to following the logic path beginning at Block 615 of Fig. 6. Fig. 7 provides a visual representation of logic used by the SingleLinkHelperImpl to set a single association end, with annotations to illustrate this particular example. Figs. 11 and 12 depict sample XMI documents representing the effect of this operation. At Block 705, the operation of setting the employee’s department begins. Block 710 checks the inverse association end to see if the employee is already linked to a department. If so, which is the case in the example (see 1120 of Fig. 11), then that link is disconnected in Block 715, thereby removing the employee from the department’s list of employees. (Note that this is not a referential integrity constraint violation, because the department’s employees association is zero to many.) In the example, this corresponds to removing the XMI syntax which defines employee E003 as having an employee link from the Shoes department, with the result as shown at 1220 of Fig. 12.

The employee’s department link is then temporarily set to null (Block 730). In Block 725, the inverse association end is set by adding the employee to the new department’s list of employees. Thus, XMI syntax is added to the XMI document 1200 to define employee E003 as having an employee link from the Clothing department, as shown at 1210 of Fig. 12. Next, the association end is modified by setting the employee’s department link to the new Clothing

value (Block 720), thereby completing the process (as represented by the resulting XMI document 1200 in this example).

Note that while the employee temporarily has no associated department as the processing of Fig. 7 progresses, in violation of the one-to-one referential integrity constraint, the operations of Blocks 730, 725, and 720 are preferably performed in an atomic manner, thus ensuring that no violations remain once the processing is complete. This treatment of operations as atomic is well known in the art.

DRAFT - 10 - 15 - 20 - 25 - 30 - 35 - 40 - 45 - 50 - 55 - 60 - 65 - 70 - 75 - 80 - 85 - 90 - 95

Block 715 of Fig. 7 corresponds to Block 625 of Fig. 6, whereby in order to set John Smith's department to Clothing, his department (inverse) association end is first removed (such that the Shoes department no longer includes John Smith). Block 725 of Fig. 7 corresponds to Block 635 of Fig. 6, whereby the inverse association end – i.e. the employee association from the Clothing department – is automatically and programmatically set by the present invention, ensuring that this employee association is the proper inverse of the department association which is being established by the SingleLinkHelperImpl. The Clothing department therefore includes John Smith at this point. Finally, Block 720 of Fig. 7 corresponds to Block 650 of Fig. 6, where the department association end is set to the desired value, giving John Smith a department association to the Clothing department.

Continuing with the example to illustrate how the present invention operates, suppose an employee named “Jane Doe” is to be added to the list of employees held by the Clothing

department. This is a many relationship (i.e. a department may have many employees), and thus the logic path beginning at Block 620 of Fig. 6 is followed. Fig. 8 provides a visual representation of logic used by the ManyLinkHelperImpl to set a many association end, and has annotations to illustrate this particular example. Figs. 13 and 14 depict sample XMI documents representing the effect of this operation. In the “before” XMI document 1300, Jane Doe (who is employee “E005”, as indicated at 1020 of Fig. 10) is shown at 1320 as being in the list of the employees of the Shoes department, while only employee E003 is in the list of employees of the Clothing department (see 1310). At Block 805, the operation of adding Jane Doe to the list of the Clothing department’s employee associations begins. Block 810 adds the employee to the employee list. At this point, Jane Doe appears as an employee of both the Clothing and Shoes departments. Block 820 then checks to see if the employee is already linked to a department. If so, which is the case in the example (see 1320 in Fig. 13), then that link is disconnected in Block 830, thereby removing the employee from the department’s list of employees. In the example, this corresponds to removing the XMI syntax 1320 which defines employee E005 as having an employee link from the Shoes department, leaving her as an employee of the Clothing department. (From the perspective of the employees association end, no referential integrity constraints are violated by the process of adding Jane Doe to the Clothing department while she is linked to the Shoes department, nor when removing her from the Shoes department’s employee list.)

20 Next, the employee’s department (inverse) association is set to null in Block 825, and this inverse association end for Jane Doe will then be set to point to the new department (i.e.

Clothing) in Block 830, thereby completing the process. Thus, the resulting XMI document 1400 in Fig. 14 for this example shows that the Clothing department now has links to two employees (see 1410 and 1420), while the Shoes department has no links to employees (see 1430).

5 While the employee temporarily has no associated department as the processing of Fig. 8 progresses, in violation of the one-to-one referential integrity constraint, the operations of Blocks 825 and 815 are preferably performed in an atomic manner, thus ensuring that no violations remain once the operation is finished.

10 Block 810 of Fig. 8 corresponds to Block 630 of Fig. 6, whereby in order to add Jane Doe as an employee of the Clothing department, an employee association end is created for her in that department (such that the Clothing department now includes Jane Doe). Block 825 of Fig. 8 corresponds to Block 640 of Fig. 6, whereby the existing inverse – i.e. the department association end – for this employee is removed. Finally, Block 815 of Fig. 8 corresponds to Block 655 of Fig. 6, where the inverse association end is automatically and programmatically 15 maintained by the present invention to ensure that Jane Doe has a department association, and that this department association is the proper inverse of the employee association which is being established by the ManyLinkHelperImpl.

In an optional enhancement, only the association end which has been designated as the primary end is serialized during a serialization operation. The “isPrimary” attribute of a

LinkHelperImpl preferably returns a Boolean, indicating whether or not this is the primary end. For an end that is not primary, no serialization needs to be done. For the primary end, the serialization proceeds as in the prior art. Preferably, an algorithmic approach is used to determine which end is the primary end. An algorithm used by preferred embodiments to evaluate a particular association end is presented in Fig. 15. Note that the isPrimary flag is not set at the end of the algorithm if the association end is not navigable.

As has been demonstrated, the present invention provides an efficient technique for programmatically enforcing referential integrity constraints when modifying links between class instances which are specified according to an arbitrary modeling framework. The inverse links are programmatically maintained, relieving the application programmer of a significant burden in terms of time and complexity for the code to be written (and subsequently maintained) for a particular application. The amount of data to be serialized is also reduced when the optional enhancement is implemented, and the process of maintaining referential integrity among serialized instances is improved because redundant information about the values of the same association will not have to be written across different documents or storage repositories.

While a preferred embodiment of the present invention has been described, additional variations and modifications in that embodiment may occur to those skilled in the art once 20 they learn of the basic inventive concepts. Therefore, it is intended that the appended claims shall be construed to include both the preferred embodiment and all such variations and such

modifications as fall within the spirit and scope of the invention.